UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/082,776	02/25/2002	Victor A. Bennett	BENNETT 6-5	4410

47396 7590 05/06/2005

HITT GAINES, PC
AGERE SYSTEMS INC.
PO BOX 832570
RICHARDSON, TX 75083

EXAMINER	
LI, AIMEE J	
ART UNIT	**PAPER NUMBER**
2183	

DATE MAILED: 05/06/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

	Application No.	Applicant(s)
Office Action Summary	10/082,776	BENNETT ET AL.
	Examiner	**Art Unit**
	Aimee J. Li	2183

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *09 February 2005*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-42* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-42* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date _____.

4) ☐ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____ .

5) ☐ Notice of Informal Patent Application (PTO-152)

6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-42 have been considered. Claims 2, 9, and 16 have been amended as per Applicant's request.

Papers Submitted

2. It is hereby acknowledged that the following papers have been received and placed of record in the file: Amendment as filed 09 February 2005.

Claim Rejections - 35 USC § 112

3. The following is a quotation of the first paragraph of 35 U.S.C. 112:

> The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same and shall set forth the best mode contemplated by the inventor of carrying out his invention.

4. Claims 2, 9, and 16 are rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the enablement requirement. The claim(s) contains subject matter which was not described in the specification in such a way as to enable one skilled in the art to which it pertains, or with which it is most nearly connected, to make and/or use the invention. Examiner could not determine where in the specification an apparatus and/or method for *preventing* a thread from executing until a device request is fulfilled (Applicant's claim 1) while modifying the thread to *allow* said thread to continue to traverse the pipeline while waiting for said device request to be fulfilled (Applicant's claim 2). When an instruction traverses a pipeline, it is being executed.

5. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

6. Claims 2, 9, and 15 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which

applicant regards as the invention. It is unclear how an apparatus and/or method for *preventing* a thread from executing until a device request is fulfilled (Applicant's claim 1) can also modify the thread to *allow* said thread to continue to traverse the pipeline while waiting for said device request to be fulfilled (Applicant's claim 2). When an instruction traverses a pipeline, it is being executed.

Claim Rejections - 35 USC § 102

7. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

8. Claims 1-3, 6-10, 13-14 are rejected under 35 U.S.C. 102(b) as being taught by Parady, U.S. Patent Number 5,933,627 (herein referred to as Parady).

9. Referring to claim 1, Parady has taught a context switching system for a multi-thread execution pipeline loop having a pipeline latency, comprising:

a. A context switch requesting subsystem configured to:

i. Detect a device request from thread executing within said multi-thread execution pipeline loop for access to a device having a fulfillment latency exceeding said pipeline latency (Parady Abstract; column 1, lines 29-35 and 46-57; column 2, lines 18-34; column 3, line 57 to column 4, line 18; column 4, lines 42-62; Figure 3), and

ii. Generate a context Switch request for said thread (Parady Abstract; column 1, lines 29-35 and 46-57; column 2, lines 18-34; column 3, line 57 to column 4, line 18; column 4, lines 42-62; Figure 3); and

b. A context controller subsystem configured to receive said context switch request and prevent said thread from executing until said device request is fulfilled (Parady Abstract; column 1, lines 29-35 and 46-57; column 2, lines 18-34; column 3, line 57 to column 4, line 18; column 4, lines 42-62; Figure 3).

10. Referring to claim 8, Parady has taught for use with a multi-thread execution pipeline loop having a pipeline latency, a method of operating a context switching system, comprising:

a. Detecting a device request from a thread executing within said multi-thread execution pipeline loop for access to a device having a fulfillment latency exceeding said pipeline latency (Parady Abstract; column 1, lines 29-35 and 46-57; column 2, lines 18-34; column 3, line 57 to column 4, line 18; column 4, lines 42-62; Figure 3);

b. Generating a context switch request for said thread when said thread issues said device request (Parady Abstract; column 1, lines 29-35 and 46-57; column 2, lines 18-34; column 3, line 57 to column 4, line 18; column 4, lines 42-62; Figure 3); and

c. Receiving said context switch request and preventing said thread from executing until said device request is fulfilled (Parady Abstract; column 1, lines 29-35 and 46-57; column 2, lines 18-34; column 3, line 57 to column 4, line 18; column 4, lines 42-62; Figure 3).

11. Referring to claims 2 and 9, Parady has taught wherein said context controller subsystem is further configured to modify said thread to allow said thread continue traverse said multi-thread execution pipeline loop while waiting for said device request to be fulfilled (Parady column 4, lines 53-62).

12. Referring to claims 3 and 10, Parady has taught wherein said context controller subsystem is further configured to allow other threads within said multi-thread execution pipeline loop to continue to execute while said thread is waiting for said device request to be fulfilled (Parady column 2, lines 28-34; column 3, line 57 to column 4, line 18; column 4, line 42-62; and Figure 3).

13. Referring to claims 6 and 13, Parady has taught wherein said context controller subsystem *is* further configured to replace said thread's current instruction with a NO-Operation (NOP) instruction to prevent said thread from executing until said device request is fulfilled (Parady Abstract; column 1, lines 29-35 and 46-57; column 2, lines 18-34; column 3, line 57 to column 4, line 18; column 4, lines 42-62; Figure 3). In regards to Parady, when a context switches occurs between a current thread and the next thread, the current thread becomes, in essence NOPs, since the instructions from the current thread are no longer actively being executed. The current threads instructions do nothing, which is the definition of a NOP. Please see Rosenberg's Computers, Information Processing & Telecommunications Second Edition ©1987 for clarification of the definition of NOP.

14. Referring to claims 7 and 14, Parady has taught wherein said device request is a request to access external memory due to a cache miss status (Parady Abstract; column 1, lines 29-35

and 46-57; column 2, lines 18-34; column 3, line 57 to column 4, line 18; column 4, lines 42-62; Figure 3).

Claim Rejections - 35 USC § 103

15. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

16. Claims 4-5 and 11-12 are rejected under 35 U.S.C. 103(a) as being unpatentable over Parady, U.S. Patent Number 5,933,627 (herein referred to as Parady), as applied to claims 1 and 8 above, in view of Kon and Medina's Round-Robin Scheduling © 29 January 1996 (herein referred to as Kon). Parady has not explicitly taught a miss fulfillment first-in-first-out buffer (FIFO), said context controller subsystem further configured to employ said FIFO to:

a. Store said thread in said miss fulfillment FIFO upon reaching an end position of said multi-thread execution pipeline loop (Applicant's claims 4 and 11),

b. Sequence said thread through said miss fulfillment FIFO (Applicant's claims 4 and 11), and

c. Reinsert said thread into said multi-thread execution pipeline loop at a beginning position (Applicant's claims 4 and 11).

d. Wherein said context controller subsystem is further configured to store said thread in said miss fulfillment FIFO upon receiving said context switch request (Applicant's claims 5 and 12).

17. However, Parady has taught round robin scheduling is used in the switch device (Parady column 4, lines 42-46). Kon has taught a miss fulfillment first-in-first-out buffer (FIFO), said context controller subsystem further configured to employ said FIFO to:

a. Store said thread in said miss fulfillment FIFO upon reaching an end position of said multi-thread execution pipeline loop (Applicant's claims 4 and 11) (Kon "What is Round-Robin Scheduling?"),

b. Sequence said thread through said miss fulfillment FIFO (Applicant's claims 4 and 11) (Kon "What is Round-Robin Scheduling?"), and

c. Reinsert said thread into said multi-thread execution pipeline loop at a beginning position (Applicant's claims 4 and 11) (Kon "What is Round-Robin Scheduling?").

d. Wherein said context controller subsystem is further configured to store said thread in said miss fulfillment FIFO upon receiving said context switch request (Applicant's claims 5 and 12) (Kon "What is Round-Robin Scheduling?").

18. In regards to Kon, the queue is like the FIFO buffer, since it runs the oldest, i.e. the process first put in the queue, first. Round robin scheduling is simple and effective at ensuring all threads are executed, so no thread is ever "starved." A starved thread is one that never executes, so its processes are never executed and/or completed. Round robin executes each thread for a certain interval or until a switch event, such as a long latency event, occurs, and then starts the next thread in the round robin buffer. It always executes the oldest thread in the queue first, which means that the first thread in the queue is the first thread outputted from the queue. A person of ordinary skill in the art at the time the invention was made, and as shown in Kon,

would have recognized that round robin scheduling is the simplest and fairest of scheduling algorithms (Kon "What is Round-Robin Scheduling?"). Therefore, a person of ordinary skill in the art at the time the invention was made would have incorporated Kon in the device of Parady to increase simplicity and ensure fairness between threads.

19. Claims 15-17 and 20-21 are rejected under 35 U.S.C. 103(a) as being unpatentable over Parady, U.S. Patent Number 5,933,627 (herein referred to as Parady) in view of Wilford et al., U.S. Patent Number 5,509,006 (herein referred to as Wilford).

20. Referring to claim 15, Parady has taught a fast pattern processor that receives and processes protocol data units (PDUs), comprising:

a. A dynamic random access memory (DRAM) that contains instructions (Parady column 5, lines 19-22; Figure 5; and Figure 6). In regards to Parady, DRAM in a specific type of RAM and Parady shows that RAM is used in his system. Please see Rosenberg's Computers, Information Processing & Telecommunications Second Edition for more information of RAM and DRAM.

b. A memory cache that caches certain of said instructions from said DRAM (Parady column 5, lines 19-22; Figure 5; and Figure 6); and

c. An engine that employs said DRAM and said memory cache to obtain ones of said instructions (Parady Abstract; column 1, lines 29-35 and 46-57; column 2, lines 18-34; column 3, line 57 to column 4, line 18; column 4, lines 42-62; Figure 3), including:

i. A multi-thread execution pipeline loop having a pipeline latency (Parady Abstract; column 1, lines 29-35 and 46-57; column 2, lines 18-34; column 3, line 57 to column 4, line 18; column 4, lines 42-62; Figure 3), and

ii. A context switching system for said multi-thread execution pipeline loop (Parady Abstract; column 1, lines 29-35 and 46-57; column 2, lines 18-34; column 3, line 57 to column 4, line 18; column 4, lines 42-62; Figure 3), having:

(1) A context switch requesting subsystem that: detects a device request from a thread executing within said multi-thread execution pipeline loop for access to a device having a fulfillment latency exceeding said pipeline latency (Parady Abstract; column 1, lines 29-35 and 46-57; column 2, lines 18-34; column 3, line 57 to column 4, line 18; column 4, lines 42-62; Figure 3), and

(2) Generates a context switch request for said thread (Parady Abstract; column 1, lines 29-35 and 46-57; column 2, lines 18-34; column 3, line 57 to column 4, line 18; column 4, lines 42-62; Figure 3), and

iii. A context controller subsystem that receives said context switch request and prevents said thread from executing until said device request is fulfilled (Parady Abstract; column 1, lines 29-35 and 46-57; column 2, lines 18-34; column 3, line 57 to column 4, line 18; column 4, lines 42-62; Figure 3).

21. Parady has not taught a tree engine that parses data within said PUDs. Wilford has taught a tree engine that parses data within said PUDs (Wilford column 1, lines 34-42; column 1, line 65 to column 2, line 19; column 14, lines 14-35; and Figure 5B). A person of ordinary skill in the art at the time the invention was made, and as taught in Wilford, would have recognized that a tree engine that parses data within said PUDs identifies which protocol the data belongs to in order to send the data to the correct destination (Wilford column 1, lines 34-42), thereby ensuring correct data execution. Therefore, it would have been obvious to a person of ordinary skill in the art to incorporate the tree engine of Wilford in the device of Parady to ensure correct data execution.

22. Referring to claim 16, Parady has taught wherein said context controller subsystem is further modifies said thread to allow said thread continue traverse said multi-thread execution pipeline loop while waiting for said device request to be fulfilled (Parady column 4, lines 53-62).

23. Referring to claim 17, Parady has taught wherein said context controller subsystem is further configured to allow other threads within said multi-thread execution pipeline loop to continue to execute while said thread is waiting for said device request to be fulfilled (Parady column 2, lines 28-34; column 3, line 57 to column 4, line 18; column 4, line 42-62; and Figure 3).

24. Referring to claim 20, Parady has taught wherein said context controller subsystem *is* further configured to replace said thread's current instruction with a NO-Operation (NOP) instruction to prevent said thread from executing until said device request is fulfilled (Parady Abstract; column 1, lines 29-35 and 46-57; column 2, lines 18-34; column 3, line 57 to column 4, line 18; column 4, lines 42-62; Figure 3). In regards to Parady, when a context switches occurs

between a current thread and the next thread, the current thread becomes, in essence NOPs, since the instructions from the current thread are no longer actively being executed. The current threads instructions do nothing, which is the definition of a NOP. Please see Rosenberg's Computers, Information Processing & Telecommunications Second Edition ©1987 for clarification of the definition of NOP.

25. Referring to claim 21, Parady has taught wherein said device request is a request to access external memory due to a cache miss status (Parady Abstract; column 1, lines 29-35 and 46-57; column 2, lines 18-34; column 3, line 57 to column 4, line 18; column 4, lines 42-62; Figure 3).

26. Claims 18-19 are rejected under 35 U.S.C. 103(a) as being unpatentable over Parady, U.S. Patent Number 5,933,627 (herein referred to as Parady) in view of Wilford et al., U.S. Patent Number 5,509,006 (herein referred to as Wilford), as applied to claim 15 above, in further view of Kon and Medina's Round-Robin Scheduling © 29 January 1996 (herein referred to as Kon). Parady has not explicitly taught a miss fulfillment first-in-first-out buffer (FIFO), said context controller subsystem further configured to employ said FIFO to:

a. Store said thread in said miss fulfillment FIFO upon reaching an end position of said multi-thread execution pipeline loop (Applicant's claim 18),

b. Sequence said thread through said miss fulfillment FIFO (Applicant's claim 18), and

c. Reinsert said thread into said multi-thread execution pipeline loop at a beginning position (Applicant's claim 18).

d. Wherein said context controller subsystem is further configured to store said thread in said miss fulfillment FIFO upon receiving said context switch request (Applicant's claim 19).

27. However, Parady has taught round robin scheduling is used in the switch device (Parady column 4, lines 42-46). Kon has taught a miss fulfillment first-in-first-out buffer (FIFO), said context controller subsystem further configured to employ said FIFO to:

a. Store said thread in said miss fulfillment FIFO upon reaching an end position of said multi-thread execution pipeline loop (Applicant's claim 18) (Kon "What is Round-Robin Scheduling?"),

b. Sequence said thread through said miss fulfillment FIFO (Applicant's claim 18) (Kon "What is Round-Robin Scheduling?"), and

c. Reinsert said thread into said multi-thread execution pipeline loop at a beginning position (Applicant's claim 18) (Kon "What is Round-Robin Scheduling?").

d. Wherein said context controller subsystem is further configured to store said thread in said miss fulfillment FIFO upon receiving said context switch request (Applicant's claim 19) (Kon "What is Round-Robin Scheduling?").

28. In regards to Kon, the queue is like the FIFO buffer, since it runs the oldest, i.e. the process first put in the queue, first. Round robin scheduling is simple and effective at ensuring all threads are executed, so no thread is ever "starved." A starved thread is one that never executes, so its processes are never executed and/or completed. Round robin executes each thread for a certain interval or until a switch event, such as a long latency event, occurs, and then starts the next thread in the round robin buffer. It always executes the oldest thread in the queue

first, which means that the first thread in the queue is the first thread outputted from the queue. A person of ordinary skill in the art at the time the invention was made, and as shown in Kon, would have recognized that round robin scheduling is the simplest and fairest of scheduling algorithms (Kon "What is Round-Robin Scheduling?"). Therefore, a person of ordinary skill in the art at the time the invention was made would have incorporated Kon in the device of Parady to increase simplicity and ensure fairness between threads.

Response to Arguments

29. Examiner withdraws the objections to the drawings in favor of the drawing amendments.

30. Examiner withdraws the objections to the specification in favor of the specification amendments.

31. Applicant's arguments filed 09 February 2005 have been fully considered but they are not persuasive.

32. Applicant' argues in essence on pages 9-10

> ...Support for the amended claims can be found in paragraph 56 that discusses modifying a thread (*i.e.*, replacing a thread's current instruction with a NO-Operation (NOP) instruction) to allow the thread to continue to traverse a multi-thread execution pipeline loop while waiting for a device request to be fulfilled.

33. This has not been found persuasive. The specification states that the replacement of a current thread's instructions in the pipeline with NOPs is one embodiment of the invention and the language in the specification states "The context controller subsystem 524, in one embodiment, is further configured to replace the thread's current instruction with a NO-Operation (NOP) instruction to *prevent the thread from executing until the device request is*

fulfilled (page 27, paragraph 56, lines 4-8)" The language of the specification does not expand on how the thread is allowed to continue traversing the pipeline. The language actually states that the NOP *prevents* the thread from executing until the device request is fulfilled. Also, the language which relates to this claim limitation states "The context controller subsystem 524 *may also* may also allow the thread to continue to traverse the multi-thread execution pipeline loop 500 while waiting for the device request to be fulfilled (pages 27-28, paragraph 56, lines 8 to 1)" The NOP instructions is not clearly linked as related to the traversal of the pipe. In fact, the way the sentence is phrased and taken with the fact that the paragraph is basically listing different embodiments of the invention, allowing the thread to continue to traverse the pipeline seems to be a separate embodiment from the NOP instruction embodiment. In addition, these sentences are merely statements that this is present, but does not provide details on how to implement a system that prevents execution of a thread while executing the same thread at the same time, as is required by the claim language.

34. Applicant argues in essence on pages 10-13

> ...Parady does not teach, however, generating a context switch request for a thread executing within a multi-thread execution pipeline loop when the thread issues a device request for access to a fulfillment latency exceeding the pipeline latency as recited in independent Claims 1 and 9. Instead, Parady teaches switching between threads of a program **in response** to a long-latency **event**...Thus, Parady teaches switching between threads **after a miss occurs** instead of generating a request for a context switch **when** a thread issues a **device request**.

35. This has not been found persuasive. Parady teaches that when a cache miss occurs, then the data must be fetched from memory (Parady column 1, lines 29-36; column 3, lines 57-65; and column 4, lines 42-52). This is a device request, since the memory is a device and the system is requesting data from the device. Parady also states that a thread switch is because the request for data from memory is a long-latency event (Parady column 1, lines 29-36; column 3, lines 57-65; and column 4, lines 42-52). The memory request is a long-latency event, because it takes longer than the pipeline latency. In one interpretation, "pipeline latency" means the amount of time allocated for one stage of the pipeline, i.e. one clock cycle. As seen in Hennessy on page 32, one clock cycle, in an exemplary 500MHz system, is 2ns. This means that the device request needs to be longer than 2ns to meet the claim limitation. Hennessy also teaches on page 41 that the length of time it takes to request data from memory is at least 80ns. The device request of 80ns is significantly longer than the 2ns pipeline latency. Another interpretation of pipeline latency is, ideally, how long it takes for an instruction to traverse the entire pipeline, i.e. when the pipeline functions perfectly, how long it takes an instruction to traverse the entire pipeline. As seen in Hennessy on page 132, a typical pipeline is five stages long. Going back to the 500MHz system in Hennessy on page 32, this means that the entire pipeline would take, ideally, 10ns to traverse the entire pipeline, since each stage takes one clock cycle or 2ns. The memory request of 80ns still takes significantly longer. In fact, for the pipeline latency in the second interpretation of the term to take as long as the memory request, the system would need a 40 stage pipeline, which, in of itself, is impractical.

Conclusion

36. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

37. A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

38. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Aimee J. Li whose telephone number is (571) 272-4169. The examiner can normally be reached on M-T 7:30am-5:00pm.

39. If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

40. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

AJL
Aimee J. Li
29 April 2005

Eddie Ch

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100